electronics

Article

Blockchain-Based Anonymous Authentication in Edge Computing Environment

Song Liu, Yuxiang Chai, Longshuo Hui and Weiguo Wu *

School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China
* Correspondence: wgwu@xjtu.edu.cn; Tel.: +86-18706838060

Abstract: Authentication is an important requirement for the security of edge computing applications. The existing authentication schemes either frequently rely on third-party trusted authorities, leading to the security risk of user information disclosure, or have high authentication overhead, causing certain pressure on the computation and communication of lightweight terminal equipment in the edge environment. In this paper, we proposed a blockchain-based anonymous authentication scheme for edge computing environments. We first designed a blockchain-based authentication architecture to store a small number of authentication elements in the blockchain network, and provide a decentralized and trusted authentication environment to ensure device anonymity and improve the security of authentication processes. Then, an elliptic cryptographic curve-based authentication scheme is proposed. It uses the chameleon hash function to dynamically generate the authentication data according to the elements stored in the blockchain and negotiate the session key, which effectively reduces the computational overhead in the authentication process. The experimental results show that the proposed scheme achieves a secure authentication process and effectively reduces the authentication overhead by up to 43.16% compared to three state-of-the-art schemes.

Keywords: identity authentication; edge computing; blockchain; device anonymity; authentication overhead

Citation: Liu, S.; Chai, Y.; Hui, L.; Wu, W. Blockchain-Based Anonymous Authentication in Edge Computing Environment. *Electronics* **2023**, *12*, 219. https://doi.org/10.3390/electronics12010219

Academic Editors: Neal Xiong and Di Jin

Received: 24 November 2022
Revised: 22 December 2022
Accepted: 30 December 2022
Published: 1 January 2023

1. Introduction

Edge computing is a new computing model resulting from the cross-development and integration of technologies such as cloud computing, IoT, and high-speed mobile communication [1]. Edge computing development has boosted a large number of emerging areas. With the popularity of smart terminal equipment and their widespread use, it not only generates huge application data, but also brings massive amounts of data access requests [2,3]. Despite the advantages of on-demand resources, dynamic scaling and cost reduction [4], cloud platforms have difficulty in efficiently handling massive data requests from various end devices, not to mention the high responsiveness requirements of these applications [5,6]. In the edge computing environment, computing, storage, and other resources are deployed near terminal equipment, which can respond to various requests from the devices in a timely manner and provide elastic services to users, thus reducing response latency and alleviating the pressure on data transmission and computation in the cloud [7]. As a result, edge computing vigorously promotes the rapid development of emerging applications such as autonomous driving [8], smart cities [9], and smart homes [10].

At the same time, edge computing is facing many new challenges. For example, security is one of the most fundamental and important challenges [11]. Without a complete solution to application security issues, any application could suffer huge losses and serious consequences [12,13]. Authentication, as the entry point to information access and external services, provides the foundation for the security of many applications. Secure authentication is the basis for users to acquire application services and is a prerequisite for authorized user access [14]. However, the traditional centralized unilateral authentication approach is fragile and suffers from a single point of failure, internal attacks or internal spoofing, which can lead to authentication failures or even system crashes [15].

Existing authentication schemes either have security risks or are costly to ensure security. Due to the decentralized and complex characteristics of lightweight terminal equipment, the complete, mature, and centralized authentication schemes deployed in cloud computing environments [16,17] find it difficult to meet the highly trusted and efficient authentication requirements of edge computing environments. Meanwhile, existing authentication schemes in edge environments usually use digital certificates to achieve decentralized and distributed authentication [18,19], but as the authentication process often relies on third-parties, this brings about complex certificate management problems and there is still a certain risk of security leakage. Meanwhile, many studies have improved the security of the authentication process by designing more sophisticated and complex authentication mechanisms [20,21], but increased the authentication overhead and caused pressure on the request response delay, data access and computational efficiency on resource-constrained terminal equipment. In short, existing authentication schemes do not achieve a good balance between secure authentication process and high authentication efficiency.

In this paper, we propose a blockchain-based anonymous authentication; a secure and efficient distributed authentication method for the specificity of the edge computing environment, which will improve the security of authentication data and reduce the authentication overhead for edge computing applications. The main contributions of this paper are as follows.

- We propose a blockchain-based identity authentication architecture. It stores the authentication data in the blockchain network and provides a distributed and trusted authentication environment for edge computing, reducing the risk of data loss and data misuse associated with centralized storage of authentication information.
- We design an elliptic cryptographic curve-based authentication scheme. Based on the elliptic curve discrete logarithm problem, we construct a chameleon hash function and use its trapdoor feature to dynamically generate a small number of authentication elements so that terminal equipment can reduce computational overhead in the process of mutual authentication.
- We analyze the security of the proposed scheme, and evaluate the computational overhead and the communication overhead of our scheme by comparing with three state-of-the-art authentication schemes. The results show that the proposed scheme outperforms the previous schemes.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 describes our authentication model scheme in detail. Section 4 presents the evaluation results. Section 5 concludes this paper.

2. Related Work

The current research on authentication technologies can be divided into two main areas, one is certificate-based authentication and the other is certificate-free authentication. For the certificate-based methods, blockchain technology has been applied to various scenarios in the edge computing environment for authentication, due to its advantages of decentralization, immutability and anonymity. Zhang et al. [22] combined the mature PKI (public key infrastructure) technology with blockchain technology to implement a semi-distributed authentication. PKI generates digital certificates, and manages the issuance and revocation of such certificates. Blockchain enables device anonymity. Ma et al. [18] applied blockchain technology to edge devices to serve IoT better, treating each edge device as a node to form a blockchain network, sending digital certificates through ECDSA signature messages, and implementing a decentralized authentication platform with the help of Fabric. To address the distribution and complex management of a large number of certificates, El-Hajj et al. [19] implemented a distributed PKI with the help of an ethernet to protect the communication channel between devices by pre-shared keys and eliminate the trust requirements of the current PKI for clients. The PKI needs to create a certificate used as an identity for client authentication. Such an approach seems practical for a limited number of clients, but it is inconvenient to distribute certificates for each IoT device, and there is no ability to manage such a large number of certificates.

In response to the disadvantages, such as complex management brought by digital certificates, the certificate-free methods usually adopt the elliptic cryptographic curve-based technique to achieve authentication, considering the characteristics of the edge computing environment with multiple domains and the authentication overhead of lightweight devices. Li et al. [23] designed an anonymous authentication scheme for real-time and lightweight device authentication to improve the traditional authentication key exchange by using Registration Center (RC) registry. RC not only provides a key registration service but also keeps the list of user information. The authentication process is accomplished by asking the RC to provide authentication information. However, the authentication process relies on a centralized trusted third-party RC, and keeping the list of user information increases the risk of user information leakage. Furthermore, this scheme achieves mutual authentication through complex bilinear pairing. For the distributed characteristics of the edge computing environment, Kuar et al. [24] used an elliptic curve cryptosystem, one-way hash function, and cascade operation to resist attacks, and provided a registration platform for devices based on RC. Jia et al. [20] proposed an identity-based AAKA protocol to achieve anonymity and un-traceability of user identities and single sign-on through a single registration, which increases the authentication overhead due to the use of expensive bilinear pairing operation. To adapt to edge computing as well as IoT environments, Gao et al. [25] proposed an efficient switching authentication scheme to reduce the authentication overhead of lightweight device authentication. Based on the elliptic curve cryptosystem [21], pseudo-identity keys are preassigned to each user ID without bilinear pairs, but a large number of pseudo-IDs need to be generated for each registered user. Any user needs to store the key corresponding to the pseudo-ID, which is impractical for terminal equipment with limited storage resources.

In addition, for the hottest smart car and telematics application scenarios of edge computing, a lot of work has been performed to customize the authentication schemes. Zhang et al. [26] designed an authentication scheme based on elliptic cryptographic technique and Chinese remainder theorem for smart cars that do not have ideal hardware conditions. Their authentication scheme does not use the expensive bilinear pairing operations, but they still require a centralized trusted third party, which also brings risks of centralized storage. In the field of vehicular ad-hoc networks, Tan et al. [27] adopted a certificateless aggregated signature authentication scheme, which compresses authentication information from multiple vehicles on the road into a single message. It can effectively reduce the computational overhead and storage overhead of roadside service units while avoiding the privacy leakage problem that may result from centralized storage. However, the certificateless aggregated signature scheme mainly benefits from the dense nature of road vehicles and roadside service units, while other edge computing application scenarios generally do not have such hardware availability.

From the existing research work, it can be seen that the certificateless authentication scheme has certain advantages over the PKI scheme in terms of security and efficiency in the edge computing scenario. However, certificate-free authentication schemes still have the risk of information leakage by relying on third parties, and the schemes with high security often bring expensive computational overhead. Moreover, in specific application scenarios, such as telematics, an efficient authentication scheme has certain requirements for the hardware environment and is not universal. For the more general edge computing application scenarios, existing authentication schemes still need to explore a balanced and effective solution in terms of security and computational efficiency. In this paper, we propose a blockchain-based certificate-free authentication scheme that uses an elliptic cryptographic curve technique to dynamically generate a small number of authentication elements to reduce computational overhead, and ensure authentication security and meet the requirements of anonymity and distributed trustworthiness by storing the authentication elements through the blockchain network.

3. Methodology

This section describes the proposed authentication architecture and the process of our authentication scheme, including system initialization, registration, and mutual authentication.

3.1. Authentication Architecture

Figure 1 shows the infrastructure architecture for authentication. The main entities include the terminal equipment (U), the key generation server/center (KGC), the edge server (ES), and the blockchain network.

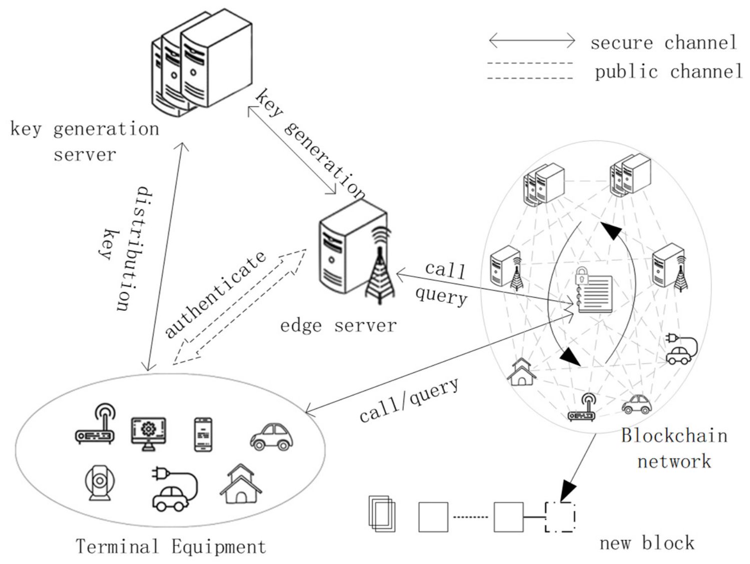


Figure 1. Authentication architecture diagram.

The blockchain network plays an important role in the proposed authentication architecture. The blockchain serves as a distributed, trusted and shared ledger, and it can securely store some of the authentication elements to enable distributed and trusted sharing of authentication data, and also provides trust help for identity authentication. The blockchain execution engine calls smart contracts and other functions, and the blockchain network makes consensus on the generated registration and other data and saves them in the blockchain.

Terminal equipment can be any device that needs to request edge-side services, such as smartphones, intelligent vehicles, smart cameras, smart meters, and other smart devices. The terminal equipment users send information content such as registration and authentication requests to the blockchain network. The requests are sent through the call interface.

The edge server can provide efficient data processing and analysis to serve terminal equipment. Each ES joined to the blockchain network can ensure the normal operation of the blockchain distributed ledger, and also communicates with the cloud to guarantee the high efficiency and low cost of application services with the help of richer computing, storage and other resources in the cloud. Terminal equipment and edge servers must register on the blockchain before obtaining or providing services.

ESs and U register on the blockchain by invoking smart contracts, and invoke smart contracts to query the authentication information and other elements of the device. The information of U is not saved in the blockchain in plaintext, but only the relevant information elements are stored to avoid revealing the identity information of the terminal equipment. In order to take advantage of the proximity of the ES to the data source to provide convenient services, there is no need to provide anonymous services for the ES. Therefore, the ES's information is stored in the blockchain in plaintext. Although the devices communicate in the blockchain in an anonymous way, the transaction information occurring at the anonymous public key address can be analyzed and, thus, there is a risk of device information leakage. Therefore, the anonymous feature of the blockchain is not used in the identity authentication scheme, but we use discrete logarithmic puzzles to achieve the anonymity of the device information.

The KGC is deployed in the cloud service. The cloud is a secure and trusted existence relative to the edge. The secure and reliable KGC can generate a large amount of random and secure key information, and is responsible for distributing key information for terminal equipment and edge servers. It is worth noting that in the proposed architecture, KGC does not save the registration information of terminal equipment and edge servers, to reduce the risk of device information leakage due to centralized storage.

The device requests key information through a secure channel during the registration and authentication process. The terminal equipment and the edge server complete the mutual authentication process through a public channel, and negotiate a secure session key to protect the transmitted data to a certain extent.

Figure 2 shows the hierarchical structure of the proposed authentication scheme. The elliptic curve encryption system (ECES) is employed to generate the authentication information. The interfaces in Figure 2 correspond to the three main phases of the authentication scheme. The edge computing applications invoke the three interfaces in turn to establish the secure connection between U and ES if it is feasible. The initialization phase initializes the ECES parameters and the public and private key parameters for the KGS. In the registration phase, U and ES separately apply for keys from KGC and are registered to the blockchain network constructed by the development tool Ganache [28]. In the mutual authentication phase, U and ES communicate with each other according to TCP protocol and socket, and access part authentication elements by querying the smart contract of the constructed blockchain network. We will describe in detail the implementation process of these three phases for the whole scheme in the next subsections.

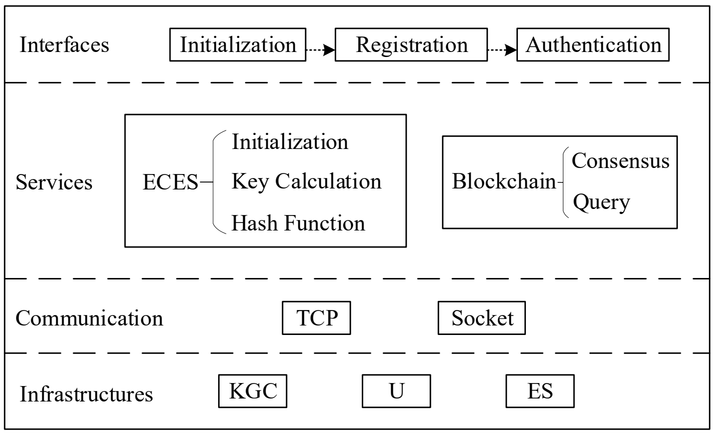


Figure 2. Hierarchical structure of the proposed authentication scheme.

3.2. Initialization Phase

The initialization phase is mainly to initialize the parameters of the elliptic curve cryptosystem and construct the public and private key information. Section 3.2.1 describes the initialization phase in detail. Section 3.2.2 describes the construction of the chameleon hash function used in the elliptic curve cryptosystem in detail.

3.2.1. System Initialization

(1) KGC first generates elliptic curves E of order q, element P, the cyclic additive group G, and the large integer group Z_q^* of order r.

(2) KGC generates random number SK_{kgc} from Z_q^* as the private key, then calculates $PK_{kgc} = SK_{kgc} \cdot P$, and constitutes its public and private key information (PK_{kgc}, SK_{kgc}), where $SK_{kgc} \in Z_q^*$.

(3) Select a chameleon hash function $H_2 : CH(g, h, r, t)$, three one-way secure hash functions that $H_1 : \{0,1\}^* \to Z_q^*$, $H_4 : \{0,1\}^* \times G \to Z_q^*$, $H_3 : G \times \{0,1\}^* \times G \times \{0,1\}^* \to Z_q^*$.

(4) $\left\langle E, G, P, q, H_1, H_2, H_3, H_4, PK_{kgc} \right\rangle$ are disclosed as a public parameter. SK_{kgc} is confidential and not disclosed.

Table 1 describes the symbolic parameters used in this paper.

Table 1. Descriptions of symbolic parameters.

Symbolic Parameters	Descriptions
PK_{kgc}	public key of key generation center
SK_{kgc}	private key of key generation center
PK_u	public key of terminal equipment
SK_{u}	private key of terminal equipment
PK_s	public key of edge server
SK_s	private key of edge server
ID_{u}	identity of terminal equipment
ID_s	identity of edge server
$SessionKey_{\mathrm{u\_s}}$	session key generated by terminal equipment
$SessionKey_{\mathrm{s}\_u}$	session key generated by edge server
E	elliptic curve
P	order of group G
G	an additive cyclic group
q	large prime numbers
Z_{q}^*	a large integers group
r	order of group Z_q^*
$H_1 : \{0,1\}^* \to Z_q^*$	hash function: used to generate fake identity
$H_2 : CH(g, h, r, t)$	chameleon hash function
$H_3 : G \times \{0,1\}^* \times G \times \{0,1\}^* \to Z_q^*$	hash function: used to generate session key
$H_4 : \{0,1\}^* \times G \to Z_q^*$	hash function: generate authentication data

3.2.2. Construction of the Chameleon Hash Function

In the authentication scheme, the elliptic curve discrete logarithm difficulty problem is used to construct the chameleon hash function. First, the terminal equipment generates a random value SK_u from the group of integers as the private trapdoor key, $SK_u \in Z_q^*$, and the group G generates a trapdoor to create the public key g. The public trapdoor key h is obtained by computing $h = g^{SK_u}$.

A random value r is generated at time T_u, where $r \in Z_q^*$. The value of the chameleon hash function T_u is obtained by calculating H_2. The construction and calculation of H_2 is given by Equation (1).

$$H_2 : CH(g, h, r, T_u) = (g^{T_u}) \cdot h^r = g^{T_u + (SK_u) \cdot r} \tag{1}$$

Since changing the random value r will definitely lead to a change in the chameleon hash function value, the hash value of the current time T_n is calculated by Equation (2) when T_u becomes T_n. In order to make the hash function value of time T_n and that of time T_u the same, i.e., Equation (3) holds, a new random value r' can be calculated by Equation (4). So far, the terminal equipment can fake different time according to the corresponding random value, and thus generate the same hash value.

$$CH(T_n) = (g^{T_n}) \cdot h^{(r')} = g^{T_n + (SK_u) \cdot (r')} \tag{2}$$

$$CH(T_u) = CH(T_n) \Rightarrow g^{T_u + (SK_u) \cdot (r)} = g^{T_n + (SK_u) \cdot (r')} \tag{3}$$

$$r' = forge(SK_u, r, T_n, T_u) = \frac{T_u - T_n}{SK_u} + r \quad (4)$$

3.3. Registration Phase

In the registration phase of the proposed scheme, the terminal equipment and the edge server first submit the key application request to the KGC separately. Second, the KGC generates key information for the device after verifying the correctness of the request information, and does not save the key information. Finally, the device invokes a smart contract to register on the blockchain network.

The terminal equipment and the edge server are registered on the blockchain network before they start authenticating with each other, which in part improves the security of the authentication process.

3.3.1. Terminal Equipment Registration

The sequence diagram of registration process on the terminal equipment is shown in Figure 3. The specific processing steps are described as follows.

(1) The terminal equipment U hashes the unique identity ID_u through $P_{id} = H_1(ID_u)$ to generate a pseudo-identity P_{id} and sends the pseudo-identity P_{id} to the KGC through a secure channel. After receiving the key request message P_{id} from the terminal equipment, KGC generates a random number r_i as the temporary private key, where $r_i \in Z_q^*$, and gets the temporary public key R_i by computing $(r_i) \cdot P$. It gets SK_u and uses it as the private key of the device by computing $r_i + (SK_{kgc}) \cdot P_{id}$. It gets the public key PK_u of the terminal equipment by computing $(SK_u) \cdot P$ and generates the public and private key information of the device. KGC generates a random number g from G as the generated public key of the chameleon hash function, gets the public key of the terminal equipment by calculating $h = g^{SK_u}$ as the public key of the trapdoor of the terminal equipment, and generates the current timestamp T_{kgc}. KGC sends the information $\left\langle SK_u, PK_u, h, T_{kgc}, R_i \right\rangle$ to the terminal equipment.

(2) After receiving the message, the terminal equipment verifies the timestamp and public key information. First, device checks if the timestamp is within the acceptable range, if not, it rejects the response message. By calculating $R_i + (PK_{kgc}) \cdot P_{id}$ and comparing it with PK_u, the response message is rejected if they are not equal.

(3) The terminal equipment generates the current timestamp T_u, along with a random number r, where $r \in Z_q^*$, and generates the chameleon hash value by $CH(g, h, r, T_u)$, and invokes smart contract $registerU(CH_{ID_u}, P_{id})$ to register the information $\langle CH_{ID_u}, P_{id} \rangle$ into the blockchain.

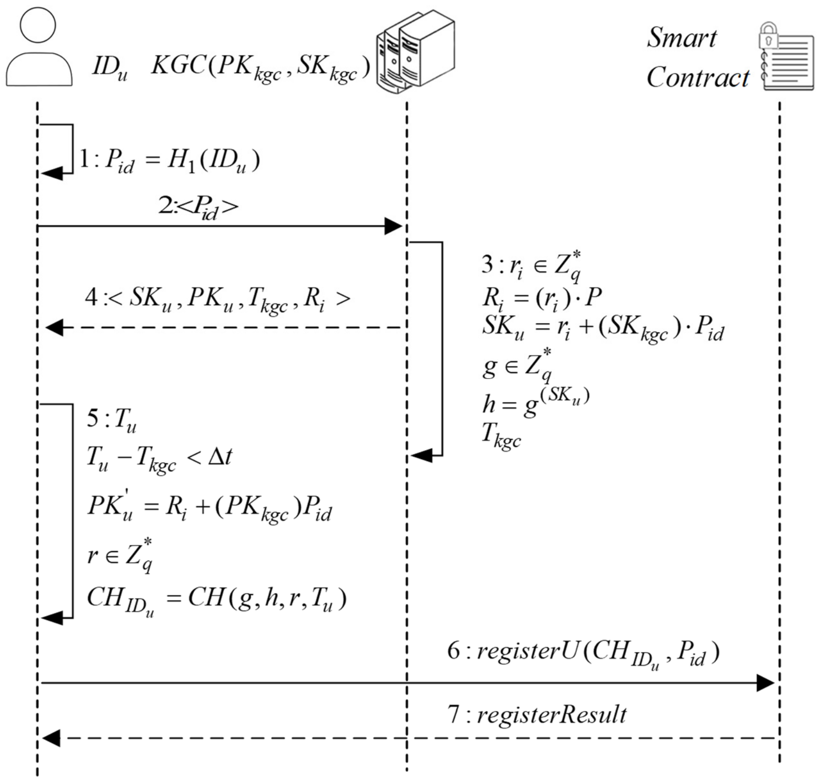
3.3.2. Server Registration

The sequence diagram of registration process on the edge server is shown in Figure 4. The specific processing steps are described as follows.

(1) The edge server ES hashes the identity ID_s, and then sends it to the KGC via a secure channel.

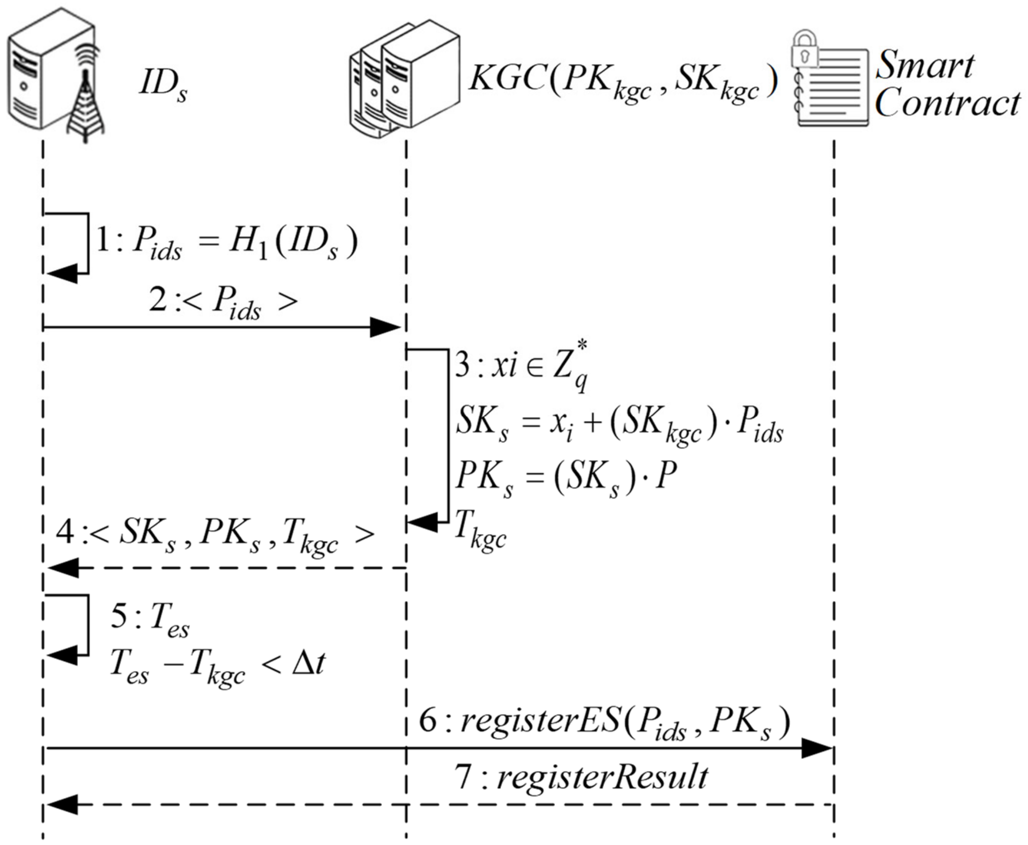
(2) After receiving the request, KGC generates a random value x_i, where $x_i \in Z_q^*$. The private key SK_s of the edge server is obtained by calculating $x_i + (SK_{kgc}) \cdot P_{ids}$. The public key PK_s is obtained by calculating $(SK_s) \cdot P$. The current timestamp T_{kgc} can be obtained, and the information $\left\langle SK_s, PK_s, T_{kgc} \right\rangle$ is sent to the edge server.

(3) After receiving the response information, the edge server first determines whether the difference between the current timestamp T_{es} and the received timestamp T_{kgc} is within the valid range. The edge server rejects the response if the difference is not within the valid range. Then, the smart contract $registerS(P_{ids}, PK_s)$ is invoked to register the public key information of the edge server into the blockchain.



ID_u
$KGC(PK_{kgc}, SK_{kgc})$
Smart Contract
$1: P_{id} = H_1(ID_u)$
$2: <P_{id}>$
$3: r_i \in Z_q^*$
$R_i = (r_i) \cdot P$
$SK_u = r_i + (SK_{kgc}) \cdot P_{id}$
$g \in Z_q^*$
$h = g^{(SK_u)}$
T_{kgc}
$4: < SK_u, PK_u, T_{kgc}, R_i >$
$5: T_u$
$T_u - T_{kgc} < \Delta t$
$PK_u^{'} = R_i + (PK_{kgc})P_{id}$
$r \in Z_q^*$
$CH_{ID_u} = CH(g, h, r, T_u)$
$6: registerU(CH_{ID_u}, P_{id})$
$7: registerResult$

Figure 3. Sequence diagram of terminal equipment registration.



ID_s
$KGC(PK_{kgc}, SK_{kgc})$
Smart Contract
$1: P_{ids} = H_1(ID_s)$
$2: < P_{ids} >$
$3: xi \in Z_q^*$
$SK_s = x_i + (SK_{kgc}) \cdot P_{ids}$
$PK_s = (SK_s) \cdot P$
T_{kgc}
$4: < SK_s, PK_s, T_{kgc} >$
$5: T_{es}$
$T_{es} - T_{kgc} < \Delta t$
$6: registerES(P_{ids}, PK_s)$
$7: registerResult$

Figure 4. Sequence diagram of edge server registration.

3.4. Authentication Phase

In the mutual authentication phase, the terminal equipment makes an authentication request to the edge server, the terminal equipment and the edge server query the relevant authentication elements by invoking the smart contract. The authentication phase invokes smart contracts to obtain authentication elements to dynamically construct authentication data. It completes the negotiation of session keys, through one round of communication, that are known only to the communicating parties.

The terminal equipment and the edge server complete the authentication by querying a part of the authentication elements stored on the blockchain network instead of KGC. The use of blockchain technology simplifies communication in the authentication process and avoids the risk of device information leakage due to centralized storage.

The sequence diagram of mutual authentication process is shown in Figure 5. The specific processing steps are described as follows.

(1) First, the terminal equipment requests access to the edge server by invoking the smart contract to obtain the public key information of ES PK'_s. Then, the terminal equipment generates a random number r' that can form the same hash value by taking the trapdoor private key SK_u, the previous random number r, the timestamp T_u, and the present timestamp T_n to calculate $h' = h^{(r')}$.

(2) The terminal equipment generates the temporary private key N_1, where $N_1 \in Z_q^*$, and calculates $(N_1) \cdot P$ to obtain the temporary public key P_1. By calculating $PK_u + (P_1) \cdot P_{id}$, the pseudo-identifier ($PIDI$) is obtained, and $\langle h', P_1, PIDI, T_n \rangle$ is sent to the edge server ES through the public channel.

(3) ES obtains the value of the chameleon hash function of the terminal equipment by computing $(h') \cdot g^{T_n}$, and invokes the smart contract to query whether the value exists. It obtains the corresponding pseudo-identity P_{id}. The public key information of U is obtained by computing $PK'_u = PIDI - (P_1) \cdot P_{id}$.

(4) ES generates a random number N_2 as the temporary private key, where $N_2 \in Z_q^*$, and calculates $(N_2) \cdot P$ to obtain the temporary public key P_2. Then, it calculates $M = (N_2) \cdot P_1$, $K = H_4(M||PK_s)$, $SessionKey_{s\_u} = H_3(PK_s||M||PK'_u||T_n)$, and sends the message $\langle P_2, K \rangle$ to U.

(5) After receiving the information from ES, the device calculates $M' = (P_2) \cdot N_1$, generates $K' = H_4(M'||PK'_s)$, and confirms the relationship between K and K'. In a case of dissimilarity, it needs to stop the authentication. After successful authentication, the session key between the terminal equipment and the edge server is calculated to obtain $SessionKey_{u\_s} = H_3(PK'_s||M'||PK_u||T_n)$.

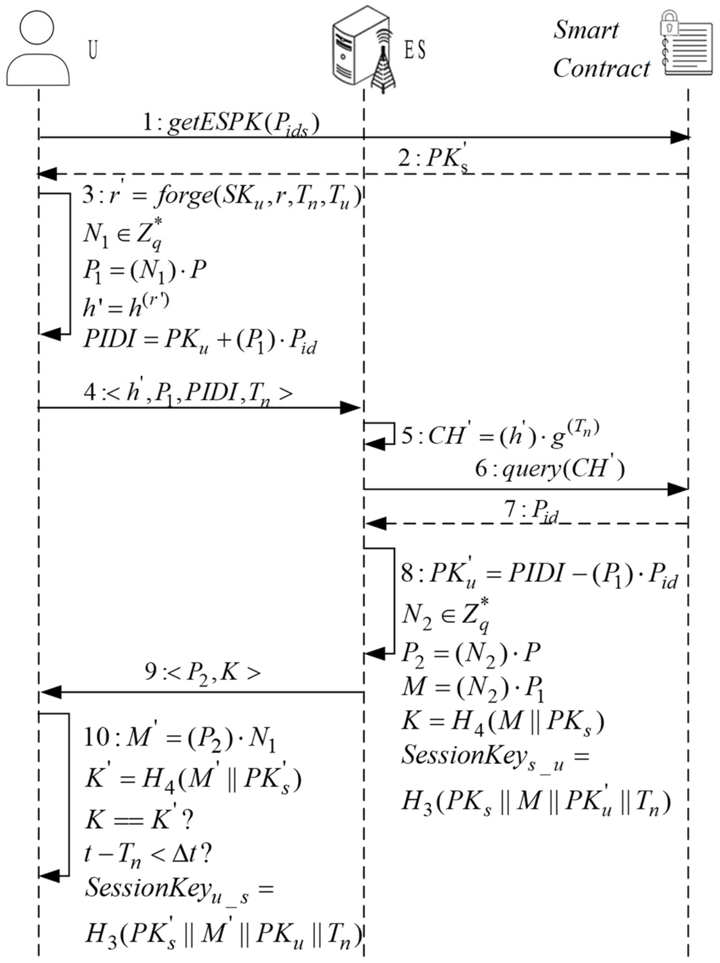


Figure 5. Sequence diagram of mutual authentication.

4. Performance Evaluation

We evaluate the proposed authentication scheme in terms of both security and authentication overhead.

4.1. Security Analysis

Scyther [29] is an automated tool for security validation, which verifies the protocol through an unlimited number of sessions and random numbers. An attacker verifies that the protocol satisfies the stated security properties without knowing the communicating parties' keys. The security properties of the protocol include Secret, Alive, Weakagree, Niagree (Non-injective agreement) and Nisynch (Non-injective Synchronization). After making a security declaration for a protocol, Scyther checks whether the protocol meets the security declaration.

We use the Scyther tool to verify the security of the mutual authentication process, assuming that the information obtained from the blockchain is secure. The results of Scyther verification show that there are no potential attack paths in the mutual authentication phase. The protocol can satisfy the session key security, and both communicating parties satisfy the declared security properties, and the protocol is resistant to man-in-the-middle attacks and replay attacks.

We also analyze the proposed authentication scheme in detail to prove that it meets the necessary security requirements [30], which include forward security, untraceable, anonymity, etc.

(1) Mutual authentication. Mutual authentication is based on K and K', both of which are dependent on M and M', that depend not only on the temporary public key P_1 and

P_2 during the message transmission conversation, but also on the public key PK_s of the edge server and the temporary private key information N_1 and N_2. Since N_1 and N_2 are not transmitted on the public channel, attackers cannot eavesdrop, steal, or analyze them. Due to the elliptic curve discrete logarithm problem (ECDLP) puzzle, attackers cannot extract the private key information from the public key information, and therefore cannot forge the authentication message.

(2) Distributed trustworthiness of the authentication process. The KGC key generation server only participates in the issuance of key information and does not store the information. The authentication process of the device does not rely on a centralized trusted third party, and thus it meets the distributed trustworthiness requirements.

(3) Forward security. Through one round of communication between the terminal equipment and the edge server, negotiating session keys $SessionKey_{s\_u} = H_3(PK_s||M||PK'_u||T_n)$ and $SessionKey_{u\_s} = H_3(PK'_s||M'||PK_u||T_n)$ are both obtained in real-time, depending on M, M' and T_n. There is no connection with the session keys to which the previous communication established a connection. The leakage of the master keys PK_s and PK_u cannot access the session keys established between devices in the past. Therefore, there is no threat to the completed communication, since no third party can access each session of the device to generate of temporary private N_1 and N_2. In addition to that, an attacker needs to know $(N_2) \cdot P_1$ and $(N_1) \cdot P_2$. Even if P_1 and P_2 are obtained by snooping the channel transmission information, the information of N_1 and N_2 cannot be cracked because ECDLP cannot parse out M and M'.

(4) Replay attacks. The establishment of session keys relies not only on the information transmitted over the communication channel, but also on temporary public and private keys and timestamps. Using multi-party information to prevent replay attacks, devices cannot negotiate a common session key through the communication information used to establish previous connection.

(5) Man-in-the-middle attack. In the authentication phase, the authentication information K and K' are related to the temporary public and private keys generated by establishing the session. The session key is also related to them. Suppose an attacker wants to act as a man-in-the-middle to forward messages and tamper with them. In that case, it must know the temporary key information and the private key information of the device to complete the authentication and key exchange process. Assuming that the private key of the device is safely stored, then the man-in-the-middle attack is impossible to achieve.

(6) Untraceable. During the authentication process, the data $\langle h', P_1, PIDI, T_n \rangle$ and $\langle P_2, K \rangle$ transmitted by the public channel are unrelated in each authentication process, and the key negotiated for each authentication is different. The attacker cannot analyze the similarity of communication from the data transmitted by the public channel to obtain the device information, so the session process between devices cannot be traced, which ensures the identity security of terminal equipment.

(7) Session key. The generation of session keys $SessionKey_{u\_s}$ and $SessionKey_{s\_u}$ depends not only on the public key information of both parties, but also on M, M' and the timestamp T_n. In the process of mutual authentication, even though the public key information PK_u of the terminal equipment and the public key PK_s of the edge server are known, and the timestamp T_n is stolen through the public channel, third parties and attackers cannot obtain them because M and M' are secure and the session keys are known only to the communicating parties.

(8) Anonymity of user identity. In the process of authentication between the terminal equipment and the edge server, the unique identification information ID_u of the terminal equipment is not transmitted, and the pseudo-identity $PIDI$ is also changed during each session. The attacker cannot obtain ID_u. In addition, due to the one-way nature of the hash function, ID_u cannot be obtained from the analysis of $PIDI$. The scheme can guarantee the anonymity of the device identity.

(9) Single sign-on. In the process of connecting the terminal equipment to the edge server, the edge server obtains the connection information of the device from the blockchain network and decrypts the login information. The edge server does not need to save the information of the terminal equipment in the whole process. Due to the trapdoor feature of the chameleon hash function, the terminal equipment only needs to apply for the key information once, and the random value of the chameleon hash function is changed according to the time of each request for authentication. Therefore, the terminal equipment can conveniently connect to any edge server by only registering once on the blockchain network.

4.2. Performance Analysis

The performance of the authentication scheme is evaluated by computational overhead and communication overhead. We compared the proposed authentication schemes with the other three schemes, SAI [23], LPA [24], and SEI [20]. These schemes used for comparison are state-of-the-art for anonymous protection authentication in edge computing environments, and they are all certificate-free schemes. Although we have made a detailed analysis of these schemes in the related work, we summarized the features of the comparison schemes in order to more intuitively compare the differences between our scheme and them. Table 2 shows the feature differences of these schemes.

Table 2. Features of comparison schemes.

Scheme	Reliance on Third-Party	Authentication Techniques	Main Operations	Using Blockchain
SAI [23]	high	authentication key exchange	one-way hash; bilinear paring	No
LPA [24]	low	elliptic cryptographic curve	one-way hash; scalar multiplication	No
SEI [20]	low	elliptic cryptographic curve	one-way hash; bilinear paring	No
ours	no	elliptic cryptographic curve	chameleon hash; scalar multiplication	Yes

Since the terminal equipment and edge servers have different performance overheads for the same cryptographic operations, we used two different platforms for simulation. The cloud server is used as the edge computing node (server) with Intel(R) Xeon(R) CPU of E5-2630 0 @ 2.30 GHz (Intel Corporation, Santa Clara, USA) and Google Nexus One cell phone with armeabi-v7a CPU (Google Inc, Santa Clara, USA) is used as the terminal (user).

Table 3 shows the overhead of performing cryptographic operations under different platforms, which fully characterizes the performance differences between the two sides of authentication in the edge computing environment.

Table 3. Time consumption of cryptographic operations (ms).

Operation	Description	User	Server
T_h	Hash function	0.089	0.009
T_{Gm}	Scalar multiplication	19.919	1.970
T_{Ga}	Point addition	0.118	0.012
T_{Gb}	Bilinear paring	48.66	5.275
T_e	Modular exponentiation	3.328	0.339

4.2.1. Computational Overhead Analysis

Table 4 shows the time consumption of operations related to the chameleon hash function, where the time consumption for forging random values and generating hash values are indicated, respectively.

Table 4. Time consumption of operations related to the chameleon hash function (ms).

	User	Server
operation	T_{forge}	T_{ch}
description	fake random value	generate chameleon hash
required operation	$T_e + T_{Ga}$	$T_e + T_{Ga}$
time (ms)	3.446	0.69

In the authentication process, only the computational cost of the authentication phase is estimated in [20,23,24]. The initialization and registration are generally performed only once. And the other computational overhead consumption is very small and can be ignored. In addition, the call request latency for queries and other calls to smart contracts in the blockchain does not involve the performance consumption of lightweight devices. Therefore, we do not account for the time consumption of initialization and registration into the computational overhead. We analyzed the numbers of different cryptographic operations used in the authentication process of the schemes, and Table 5 shows the numbers.

Table 5. Operations required to calculate the overhead.

	User	Server
SAI [23]	$6T_{Gm} + 4T_{Ga} + 5T_h$	$T_{Gb} + 4T_{Gm} + 3T_{Ga} + 2T_h$
LPA [24]	$4T_{Gm} + 4T_h$	$3T_{Gm} + 4T_h$
SEI [20]	$4T_{Gm} + T_e + 5T_h$	$T_{Gb} + 5T_{Gm} + 3T_{Ga} + 5T_h$
ours	$T_{forge} + 3T_{Gm} + T_{Ga} + 2T_h$	$T_e + 4T_{Gm} + T_{Ga} + 2T_h + T_{ch}$

The computational overheads of the terminal and server side are compared as a whole. We used the values from Tables 3 and 4 to compute the computational overheads of different schemes according to Table 5.

Figure 6 shows the comparison of the authentication calculation overhead on the terminal side. It is clear that the proposed scheme has the minimum calculation time on the terminal side. According to Table 5, the proposed scheme has the least number of scalar point multiplication and point addition. The proposed authentication scheme reduces the computational cost in the terminal by 44.51%, 16.50%, and 19.92%, respectively, compared to SAI, LPA, and SEI.

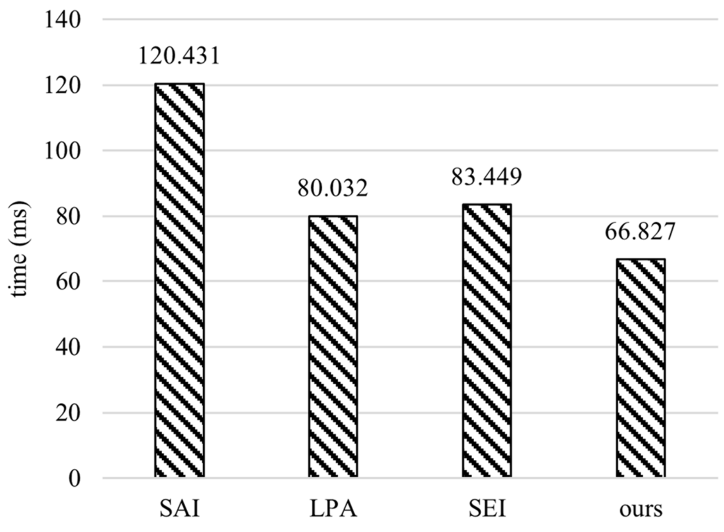


Figure 6. Computational overhead of SAI [23], LPA [24], SEI [20], and ours on the terminal side.

The comparison of the computational overhead on the edge server side is shown in Figure 7. The computational overhead of our scheme is lower than that of SAI and

SEI, both of which use the most time-consuming bilinear mapping pairing operation at the server side and, thus, they have higher time consumption. Our scheme achieves 32.33% and 41.21% lower overhead on the edge server side than SAI and SEI, respectively. However, LPA performs better than the proposed scheme. This is because LPA has less computation operations and computational overhead in the authentication process at the edge server side, according to Table 5. Although our scheme has a 33.48% higher computational overhead on the edge side than LPA, it still outperforms LPA in term of the overall overhead.

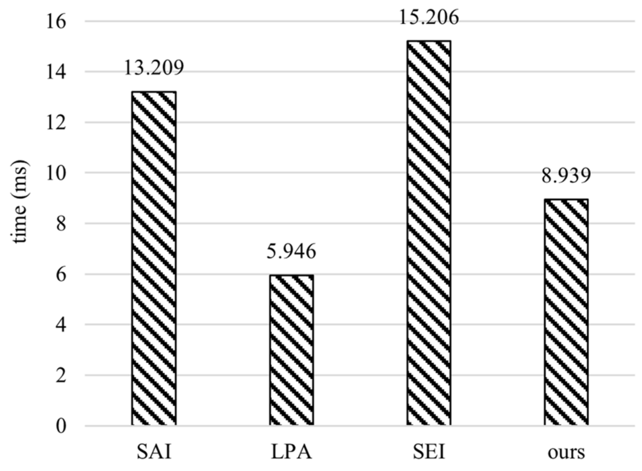


Figure 7. Computational overhead of SAI [23], LPA [24], SEI [20], and ours on the edge server.

Figure 8 shows the overall computational overheads of both terminal and server side during the authentication process. The result shows that the proposed scheme has the least overall computational overhead. In summary, its computational overhead is 43.16%, 11.64%, and 22.30% lower than that of SAI, LPA, and SEI, respectively, indicating that the proposed scheme is lightweight and requires relatively low computational overhead.

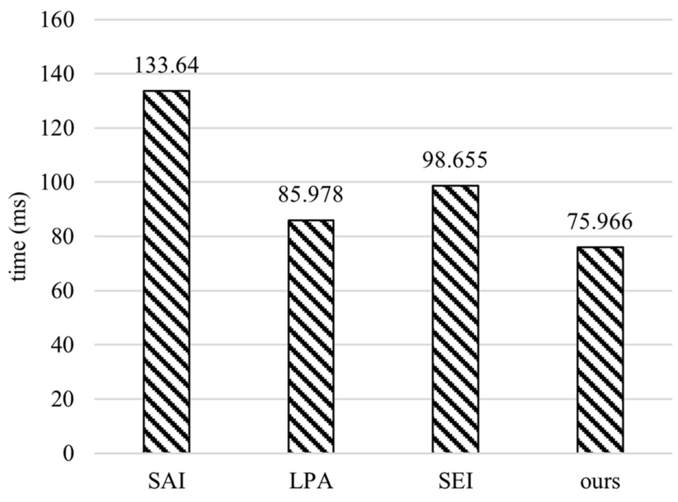


Figure 8. Overall authentication overhead of SAI [23], LPA [24], SEI [20], and ours.

4.2.2. Communication Overhead Analysis

To compare the communication overhead of the proposed scheme with other schemes, the number of messages transmitted between the participants is evaluated. These transmitted messages are represented by different symbols with different sizes, such as large integer groups Z_q^*, logos ID, hashes H, timestamps T, and cyclic addition groups G. And in the proposed authentication scheme, the sizes of these messages are set as follows:

$\left|Z_q^*\right| = 160\text{b}$, $|T| = 32\text{b}$, $|H| = 256\text{b}$, $|G| = 1024\text{b}$, $|ID| = 256\text{b}$. Based on these values, the communication costs of the schemes are calculated and summarized in Table 6.

Table 6. Communication overhead calculation.

Scheme	Communication Overhead
SAL [23]	$3\left|G\right|+4\left|Z_q^*\right|$
LPA [24]	$2\|G\|+2\|T\|+2\|H\|$
SEI [20]	$4\left|G\right|+2\left|T\right|+2\left|Z_q^*\right|+\left|ID\right|$
ours	$4\left|G\right|+\left|T\right|+\left|Z_q^*\right|$

Since the communication overhead is much smaller in time than the calculation overhead, we used the data size to present the communication overhead. Figure 9 shows the amount of data transmitted for authentication under different schemes. The communication overhead of our scheme is 4288 bits, which is lower than that of SEI by reducing 9.46%, but higher compared to SAI and LPA. This is because more data needs to be transmitted when forging the random values of the chameleon hash function in exchange for the security of anonymity. Based on the security analysis, our scheme can provide good security under different scenarios in edge computing environments. Our scheme makes a good balance between security and authentication efficiency.

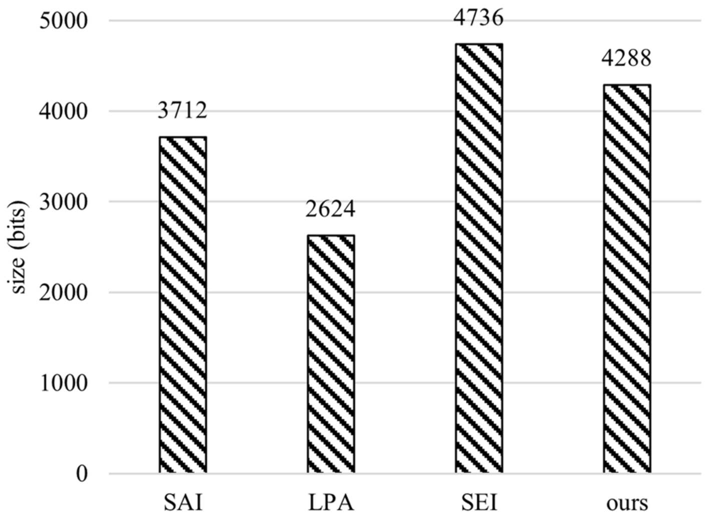


Figure 9. Communication overhead of SAI [23], LPA [24], SEI [20], and ours.

5. Conclusions

In this paper, we propose a blockchain-based authentication architecture and an effective authentication scheme for the edge computing environment. The proposed scheme provides a secure and efficient anonymous authentication for lightweight devices and edge servers. It uses the characteristics of the chameleon hash function to achieve distributed and trusted sharing of authentication data with the blockchain. We analyze the security properties and evaluate the computational and communication costs of the authentication scheme. The experimental results show that our scheme reduces the computational overhead and communication overhead to a certain extent compared with existing authentication schemes, while providing good security for authentication.

In our future work, we will try to apply the proposed authentication scheme to specific application domains, such as intelligent transportation, intelligent retail, industrial internet, intelligent medical, security monitoring, etc. For the scenarios with intensive terminal equipment, combining the aggregated authentication scheme with the proposed blockchain-based scheme to reduce the number of consensus coalescences in the blockchain would be a feasible research direction for further improving the authentication efficiency. At the same time, some methods [31,32] of providing differentiated services based on different resource allocation devices in wireless sensor networks also provide an interesting research direction for us to improve communication security and efficiency. In addition, powerful machine learning and deep learning techniques have been widely used in wireless sensor networks [33,34]. We will also explore the possibility of using AI methods in both identity authentication and blockchain consensus mechanisms for edge computing environments.

Author Contributions: Conceptualization, S.L.; methodology, S.L.; software, L.H.; validation, L.H. and Y.C.; formal analysis, S.L.; investigation, L.H.; resources, S.L.; data curation, L.H.; writing—original draft preparation, L.H.; writing—review and editing, Y.C.; visualization, Y.C.; supervision, W.W.; project administration, W.W.; funding acquisition, S.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially funded by the National Natural Science Foundation of China (No. 62002279), and Natural Science Basic Research Program of Shaanxi (Program No. 2020JQ-077).

Data Availability Statement: The data presented in this study are available in Tables 3 and 4, and Figures 6–9.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhou, C.; Yu, Y.; Yang, S.; Xu, H. Intelligent Immunity based Security Defense System for Multi-access Edge Computing Network. *China Commun.* **2021**, *18*, 100–107. [CrossRef]
2. Hegde, M.; Rao, R.R.; Nikhil, B.M. DDMIA: Distributed Dynamic Mutual Identity Authentication for Referrals in Blockchain-Based Health Care Networks. *IEEE Access* **2022**, *10*, 78557–78575. [CrossRef]
3. Li, X.; Zhang, K.; Zhang, L.; Zhao, X. A New Quantum Multiparty Simultaneous Identity Authentication Protocol with the Classical Third-Party. *Entropy* **2022**, *24*, 483. [CrossRef] [PubMed]
4. Muniswamaiah, M.; Agerwala, T.; Tappert, C. A Survey on Cloudlets, Mobile Edge, and Fog Computing. In Proceedings of the 2021 8th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2021 7th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom), Washington, DC, USA, 26–28 June 2021; pp. 139–142.
5. Zhong, H.; Zhang, C.; Xu, Y.; Liu, L. Authentication and Key Agreement based on Anonymous Identity for Peer-to-Peer Cloud. *IEEE Trans. Cloud Comput.* **2020**, *10*, 1592–1603. [CrossRef]
6. Zhou, J.; Wu, N.; Wang, Y.; Gu, S.; Cao, Z.; Dong, X.; Choo, K. A Differentially Private Federated Learning Model against Poisoning Attacks in Edge Computing. *IEEE Trans. Dependable Secur. Comput.* **2022**, accepted. [CrossRef]
7. Wum, J.; Tseng, Y.; Huang, S. An Identity-based Authenticated Key Exchange Protocol Resilient to Continuous Key Leakage. *IEEE Syst. J.* **2019**, *13*, 3968–3979.
8. Liu, S.; Liu, L.; Tang, J.; Yu, B.; Wang, Y.; Shi, W. Edge Computing for Autonomous Driving: Opportunities and Challenges. *Proc. IEEE* **2019**, *107*, 1697–1716. [CrossRef]
9. Khan, L.; Yaqoob, I.; Tran, N.; Kazmi, S.; Dang, T.; Hong, C. Edge-Computing-enabled Smart Cities: A Comprehensive Survey. *IEEE Internet Things* **2020**, *7*, 10200–10232. [CrossRef]
10. Albataineh, H.; Nijim, M.; Bollampall, D. The Design of a Novel Smart Home Control System using Smart Grid based on Edge and Coud Computing. In Proceedings of the 2020 IEEE 8th International Conference on Smart Energy Grid Engineering (SEGE), Oshawa, ON, Canada, 12–14 August 2020; pp. 88–91.
11. Yao, A.; Jiang, F.; Li, X.; Dong, C.; Xu, J.; Xu, Y.; Li, G.; Liu, X. A Novel Security Framework for Edge Computing based UAV Delivery System. In Proceedings of the 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Shenyang, China, 20–22 October 2021; pp. 1031–1038.
12. Cao, L.; Liu, Y.; Cao, S. An Authentication Protocol in LTE-WLAN Heterogeneous Converged Network Based on Certificateless Signcryption Scheme With Identity Privacy Protection. *IEEE Access* **2019**, *7*, 139001–139012. [CrossRef]
13. Jia, X.; Luo, M.; Choo, K.; Li, L.; He, D. A Redesigned Identity-Based Anonymous Authentication Scheme for Mobile Edge Computing. *IoT-J.* **2021**, *9*, 10108–10120. [CrossRef]
14. Ayotte, B.; Banavar, M.; Hou, D.; Schuckers, S. Fast Free-Text Authentication via Instance-based Keystroke Dynamics. *IEEE Trans. Biom. Behav. Identity Sci.* **2020**, *2*, 377–387. [CrossRef]
15. Gupta, D.; Ray, S.; Singh, T.; Kumari, M. Post-Quantum Lightweight Identity-based Two-Party Authenticated Key Exchange Protocol for Internet of Vehicles with Probable Security. *Comput. Commun.* **2022**, *181*, 69–79. [CrossRef]
16. Shen, S.; Wang, H.; Zhao, Y. Identity-based Authenticated Encryption with Identity Confidentiality. *Theor. Comput. Sci.* **2022**, *901*, 1–18. [CrossRef]
17. Takieldeen, A.; Abd Elkhalik, S.; Samra, A.; Mohamed, M.; Khalifa, F. A Robust and Hybrid Cryptosystem for Identity Authentication. *Information* **2021**, *12*, 104. [CrossRef]

18. Ma, Z.; Meng, J.; Wang, J.; Shan, Z. Blockchain-based Decentralized Authentication Modeling Scheme in Edge and IoT Environment. *IoT-J.* **2020**, *8*, 2116–2123.
19. El-Hajj, M.; Fadlallah, A.; Chamoun, M.; Serhrouchni, A. Ethereum for Secure Authentication of IoT using Pre-Shared Keys (PSKs). In Proceedings of the 2019 International Conference on Wireless Networks and Mobile Communications (WINCOM), Fez, Morocco, 29 October–1 November 2019; pp. 1–7.
20. Jia, X.; He, D.; Kumar, N.; Choo, K. A Provably Secure and Efficient Identity-based Anonymous Authentication Scheme for Mobile Edge Computing. *IEEE Syst. J.* **2019**, *14*, 560–571. [CrossRef]
21. Karthikeyan, S.; El-Razouk, H. Horizontal Correlation Analysis of Elliptic Curve Diffie Hellman. In Proceedings of the 2020 3rd International Conference on Information and Computer Technologies (ICICT), San Jose, CA, USA, 9–12 March 2020; pp. 511–519.
22. Zhang, P.; Jiang, C.; Pang, X.; Qian, Y. STEC-IoT: A Security Tactic by Virtualizing Edge Computing on IoT. *IoT-J.* **2020**, *8*, 2459–2467. [CrossRef]
23. Li, Y.; Cheng, Q.; Liu, X.; Li, X. A Secure Anonymous Identity-based Scheme in New Authentication Architecture for Mobile Edge Computing. *IEEE Syst. J.* **2020**, *15*, 935–946. [CrossRef]
24. Kaur, K.; Garg, S.; Kaddoum, G.; Guizani, M.; Jayakody, D. A Lightweight and Privacy-Preserving Authentication Protocol for Mobile Edge Computing. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 9–13 December 2019; pp. 1–6.
25. Gao, S.; Su, Q.; Zhang, R.; Zhu, J.; Sui, Z.; Wang, J. A Privacy-Preserving Identity Authentication Scheme based on the Blockchain. *Secur. Commun. Netw.* **2021**, *2021*, 9992353. [CrossRef]
26. Zhang, J.; Cui, J.; Zhong, H.; Chen, Z.; Liu, L. PA-CRT: Chinese Remainder Theorem based Conditional Privacy Preserving Authentication Scheme in Vehicular Ad-Hoc Networks. *IEEE Trans. Dependable Secur. Comput.* **2019**, *18*, 722–735. [CrossRef]
27. Tan, H.; Zheng, W.; Vijayakumar, P.; Sakurai, K.; Kumar, N. An Efficient Vehicle-Assisted Aggregate Authentication Scheme for Infrastructure-Less Vehicular Networks. *IEEE Trans. Intell. Transp. Syst.* **2022**, accepted. [CrossRef]
28. Ganache—Truffle Suite. Available online: https://www.trufflesuite.com/ganache/ (accessed on 11 December 2022).
29. Scyther Tool (cispa.io). Available online: https://people.cispa.io/cas.cremers/scyther/ (accessed on 11 December 2022).
30. Chen, Y.; Chen, J. CPP-CLAS: Efficient and Conditional Privacy-Preserving Certificateless Aggregate Signature Scheme for VANETs. *IoT-J.* **2021**, *9*, 10354–10365. [CrossRef]
31. Xia, F.; Hao, R.; Li, J.; Xiong, N.; Yang, L.T.; Zhang, Y. Adaptive GTS Allocation in IEEE 802.15.4 for Real-Time Wireless Sensor Networks. *J. Syst. Archit.* **2013**, *59*, 1231–1242. [CrossRef]
32. Yao, Y.; Xiong, N.; Park, J.; Ma, L.; Liu, J. Privacy-Preserving Max/Min Query in Two-Tiered Wireless Sensor Networks. *Comput. Math. Appl.* **2013**, *65*, 1318–1325. [CrossRef]
33. Cheng, H.; Xie, Z.; Shi, Y.; Xiong, N. Multi-step Data Prediction in Wireless Sensor Networks based on One-Dimensional CNN and Bidirectional LSTM. *IEEE Access* **2019**, *7*, 117883–117896. [CrossRef]
34. Gao, Y.; Xiang, X.; Xiong, N.; Huang, B.; Lee, H.; Alrifai, R.; Jiang, X.; Fang, Z. Human Action Monitoring for Healthcare Based on Deep Learning. *IEEE Access* **2018**, *6*, 52277–52285. [CrossRef]